Recovering the Position and Orientation of Free-Form Objects from Image Contours Using 3D Distance Maps

Stéphane Lavallée, *Member, IEEE*, and Richard Szeliski, *Member, IEEE*

Abstract—**The accurate matching of 3D anatomical surfaces with sensory data such as 2D X-ray projections is a basic problem in Computer and Robot Assisted Surgery. In model-based vision, this problem can be formulated as the estimation of the spatial pose (position and orientation) of a 3D smooth object from 2D video images. We present a new method for determining the rigid body transformation that describes this match. Our method performs a least squares minimization of the energy necessary to bring the set of the camera–contour projection lines tangent to the surface. To correctly deal with projection lines that penetrate the surface, we consider the minimum *signed distance* to the surface along each line (i.e., distances inside the object are negative). To quickly and accurately compute distances to the surface, we introduce a precomputed distance map represented using an *octree spline* whose resolution increases near the surface. This octree structure allows us to quickly find the minimum distance along each line using best-first search. Experimental results for 3D surface to 2D projection matching are presented for both simulated and real data. The combination of our problem formulation in 3D, our computation of line to surface distances with the octree-spline distance map, and our simple minimization technique based on the Levenberg-Marquardt algorithm results in a method that solves the 3D/2D matching problem for arbitrary smooth shapes accurately and quickly.**

Index Terms—**Registration, 3D/2D matching, octree spline, line-to-surface distance, X-ray projections, non-linear least squares minimization.**

I. Introduction

One of the classical problem in model-based vision is the estimation of the pose (i.e., the location and orientation) of a 3D object with respect to a scene described by sensory data (2D images or 3D range data). Formally, given an object in a coordinate system Ref_{3D} and sensory data such as 2D image contours in a coordinate system Ref_{sensor}, estimate the six-component vector $\mathbf{p}$ that defines the rigid body transformation $\mathbf{T(p)}$ between Ref_{sensor} and Ref_{3D}. When using 2D contours, although this problem can theoretically be solved for a single projection, in practice two or more projections are necessary to achieve a sufficiently accurate estimate.

While the method we will describe has many applications, our primary interest is in solving multi-modality medical image matching problems [1]. For instance, an important research topic in Computer and Robot Assisted Surgery is how to match 3D images such as Magnetic Resonance Imaging (MRI) or Computed Tomography (CT) scans with X-ray projections [2]. In our primary application, prior to a surgical operation, a simulated intervention procedure is performed on 3D images (MRI or CT) of the patient. Then, at the beginning of the operation, X-ray images of the patient lying on the operating table are taken from two viewpoints. The geometry of the X-ray projections is related to a robot's reference frame using standard calibration techniques [3]. To accurately perform robot assisted surgery based on the pre-operative simulation, the exact position of the patient must be determined by matching the reference system Ref_{3D} associated with the pre-operative images with the reference system Ref_{sensor} associated with the X-ray images taken in the operating room [4]. Fig. 1 illustrates this 3D/2D matching problem.

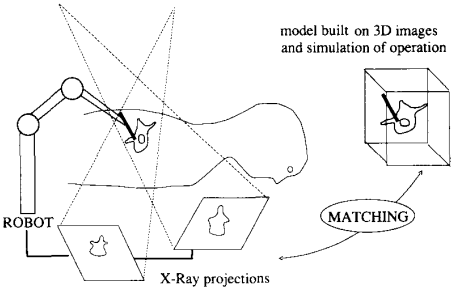


Fig. 1. In Computer and Robot Assisted Surgery, it is necessary to match 3D medical images (MRI or CT) with 2D X-Ray projections.

To perform this registration, we first segment some reference anatomical structures (e.g., vertebra for spine surgery or skull for neurosurgery) both in 3D in Ref_{3D} and in 2D in Ref_{sensor}. The 2D and 3D segmentation problems are not addressed in this paper since they constitute their own full research topics [5], [6], [7]. After the segmentation step, we match the segmented 3D smooth surface with two or more contours of the same anatomical structure extracted from its X-ray projections using our novel registration technique [8]. An

Manuscript received March 11, 1994; revised Aug. 14, 1994.

Stéphane Lavallée is with IMAG-TIMC, Faculté de Médecine de Grenoble 38 706 La Tronche Cedex, France; e-mail: stephane.lavallee@imag.fr.

Richard Szeliski is with Digital Equipment Corporation, Cambridge Research Lab, One Kendall Square, Bldg. 700, Cambridge, MA 02139; e-mail: szeliski@crl.dec.com.

IEEECS Log Number P95035.

accurate estimation of the position of the patient is thus known in the sensor coordinate system.

For such applications, the main requirements are: 1) to perform the matching process for arbitrary free-form smooth surfaces; 2) to achieve the best accuracy possible for the six rigid body transformation parameters; 3) to compute an estimate of the uncertainties in the six parameters; and 4) to perform the matching process in a reasonable time (during the surgical operation). In this paper, we focus on the 3D/2D matching problem, but a simpler version of our method can also be used to solve a 3D/3D matching problem, while satisfying the same requirements [9].

We begin the paper with a review of related work in the general fields of 3D/2D matching (Section II). We then present in Section III the new formulation of the problem that will allow us to satisfy our accuracy and speed requirements. Our method requires fast and accurate computation of line to surface distances, which we solve using 3D discrete distance maps. In Section IV, we describe a new representation of 3D distance maps based on *octree splines*. The fast computation of line to surface distances is described in Section V. Section VI presents our method for estimating the six parameters $\mathbf{p}$ of the transformation $\mathbf{T}(\mathbf{p})$ using nonlinear least squares minimization. Section VII presents some experimental results, with both synthetic data and real data. Finally, the algorithm, extensions of the method, and future improvements are discussed in Section VIII.

II. Previous Work

Estimating the pose of a 3D rigid object from sensor data has been widely studied in both medical image processing and computer vision.

In medical imaging, the problem of image registration has been usually solved using external fiducial markers placed on the body of the patient [10] or by interactively selecting pairs of matching points [11]. An automated algorithm for matching 3D surfaces with other 3D surfaces (such as the head skin surface) has been developed by Pelizzari [12]. Methods based on the registration of singular 3D curves have also been proposed in [13]. The automated matching of 3D medical images with 2D X-ray projections has not previously been studied.

In computer vision, the recovery of pose information from both 2D images and 3D range data has been widely studied. The more general problem of *elastic matching* has also received wide attention [14], [15], [16], but will not be addressed in this paper. For the method presented in this paper, we will assume that a complete representation of the 3D surface of the object is known, but will not assume anything about the existence of edges, particular points, or curves on the object.

Techniques for recovering pose from 2D images (the 3D/2D matching problem) can be classified into three general categories, based on the matching primitives used. The first class of techniques is based on matching features identified on the surface of the object [17]. The correspondence between features can be computed using either direct or coarse-to-fine procedures. These approaches resemble stereo matching algorithms and are particularly well-suited to a graph-theoretic formulation [18]. Reducing the space of the possible poses [19] and using contextual knowledge [20] can be both used to speed up the matching.

A second, more global, approach estimates the pose by aligning the extremal contours of the object with the available projections. For objects modeled as a set of polygonal surface patches, Lowe [21] describes an iterative approach for aligning the projected extremal contours with edges found in the image. His approach is similar to ours in that it uses non-linear least squares. It differs in measuring errors in the image plane and assuming a polyhedral model (which may not be suitable for complex anatomical surfaces).

For smooth objects, Besl and McKay [22] have recently proposed a very general technique (the Iterative Closest Point algorithm) for matching points to surfaces. Their method is similar to the adaptation of our 3D/2D matching algorithm presented in this paper to the 3D/3D matching problem [8], [9]. However, the authors do not address the complexity issue of rapidly finding the nearest point on a surface.

At present, there exists no efficient and general algorithm (other than our own) for the 3D/2D matching of smooth objects. The main reason for this is that inferring the contour generators (i.e., the curves that belong to the surface and that form the image contours) can be quite difficult [23]. A traditional approach is to restrict the shape of the object, e.g., to objects of revolution [24] or generalized cones [25]. Ponce and Kriegman [23] have used rational parametric surface patches, implicit algebraic equations, and elimination theory to obtain analytic expressions for the projected contours. However, their method is restricted to objects modeled by a few patches (see [26] for some recent extensions).

By comparison, our approach can deal with objects of arbitrary shape and complexity, since it does not need to explicitly compute the projected contours. In fact, our approach is similar to classical chamfer matching [27] which maximizes the correlation between 2D features in the image and the projections of 3D object features. However, our distance measure is computed in 3D, which, as we will see, removes the need to identify 3D object features.

III. Problem Formulation

In this section, we first introduce the representation and notation chosen for the rigid body transformations. We then briefly discuss the importance of sensor calibration and present our formulation of the matching problem as the minimization of an energy.

A. Transformation and Parameters

Using the notation defined in Section I, the problem is to estimate the transformation $\mathbf{T}$ between $\text{Ref}_{\text{sensor}}$ and Ref_{3D} (Fig. 2). $\mathbf{T}$ is called the *final pose* of the object, and can be defined by a translation vector $\mathbf{t}=\left(T_x, T_y, T_z\right)^T$ and a 3×3 rotation matrix $\mathbf{R}$. Several representations can be used for $\mathbf{R}$ (e.g., Euler angles, quaternions, or rotation vectors [28]). We have

chosen to use Euler angles $(\phi, \theta, \psi)^T$, where $\mathbf{R}$ is constructed from three rotations around x, y, z with angles ϕ, θ, ψ. If we gather the six parameters of the transformation $\mathbf{T}$ into a six-component vector

$$\mathbf{p} = (T_x\ T_y\ T_z\ \phi\ \theta\ \psi)^T, \tag{1}$$

and use homogeneous coordinates, $\mathbf{T}(\mathbf{p})$ can be represented by a single 4×4 matrix

$$\mathbf{T}(\mathbf{p}) = \begin{pmatrix} c_\psi c_\theta & -s_\psi c_\theta & s_\theta & T_x \\ c_\psi s_\theta s_\phi + s_\psi c_\phi & -s_\psi s_\theta s_\phi + c_\psi c_\phi & -c_\theta s_\phi & T_y \\ -c_\psi s_\theta c_\phi + s_\psi s_\phi & s_\psi s_\theta c_\phi + c_\psi s_\phi & c_\theta c_\phi & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{2}$$

where $(c_\phi, s_\phi),(c_\theta, s_\theta),(c_\psi, s_\psi)$ are the cosines and sines of the angles ϕ, θ, ψ. The Euler representation is differentiable only if some restrictions on Euler angles are applied [28]. In our case, the Euler angles remain in a single domain during the convergence because the initial estimates are not too far away from the solution.

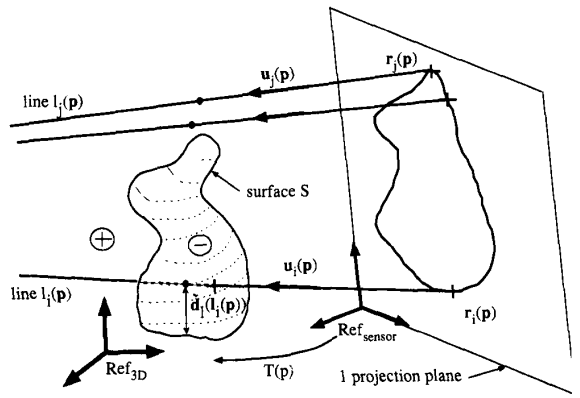


Fig. 2. Projection line to surface distance computation.

B. Relationship between 2D Image and 3D Scene Coordinates

To ensure good accuracy in the recovered parameters, accurate sensor calibration is very important [29]. Therefore, we cannot use approximations such as orthography or weak perspective (also called scaled orthography). Moreover, we must model any image distortions that deviate from the perspective model, specially for X-ray images. In order to take such *local* distortions into account with good accuracy, we have extended the two-planes method [30] to the *N-Planes Bicubic Spline (NPBS) method* [3]. By not limiting ourselves to the simplified perspective mathematical model, we can obtain an extremely accurate and simple method, using a large amount of calibration measurements and spline approximations of the calibration functions.

For each projection, two calibration planes P_1 and P_2 yield a function $C(P_i) = L_i$ that associates every pixel $P_i = (u_i, v_i)$ with a 3D line whose representation is given in $\text{Ref}_{\text{sensor}}$ by two points $P_1^i \in P_1$ and $P_2^i \in P_2$, i.e., $L_i = C(P_i) = (P_1^i, P_2^i)$. If $\mathbf{T_k}$ defines a known transformation between $\text{Ref}_{\text{sensor}}$ and a coordinate system associated with the plane P_k, the coordinates of P_1 and P_2 are given in $\text{Ref}_{\text{sensor}}$ by:

$$P_k^i = \mathbf{T_k}\,[F_{xk}(u_i, v_i), F_{yk}(u_i, v_i), 0]^T, \tag{3}$$

where the functions F_{xk} and F_{yk} are bicubic spline functions,

$$F(u, v) = \sum_{l,m} \alpha_{l,m} B_l(u) B_m(v) \tag{4}$$

B_l and B_m are B-spline functions, and $\alpha_{l,m}$ are coefficients obtained through approximation of accurate calibration data (centroids of radio-opaque spheres for X-rays or centroids of illuminated discs for video images). To simplify our notation, we represent each projection line $L_i = C(P_i) = (\mathbf{q_i}, \mathbf{v_i})$ by a point $\mathbf{q_i} = P_1^i$ and a vector $\mathbf{v_i} = \overrightarrow{P_1^i P_2^i}$ which represent the set of points $\mathbf{q_i} + \lambda \mathbf{v_i}$. In principle, we obtain a function C for each one of the N projections, but in practice, we can gather all of these N functions into one general function valid for any pixel of any projection. Note that we use the above accurate sensor model for both video cameras and X-ray projection systems.

C. Least Squares Formulation

Given N silhouette contours of an object S, we first select a set of M pixels $\{P_i,\ i = 1 \ldots M\}$ that belong to the N contours. This means that we do not need to perfectly segment the projection contours of S. Instead, only some pixels P_i that lie on the edges of the silhouette with a high certainty need to be known. Each pixel P_i corresponds to a 3D *matching line* L_i given in $\text{Ref}_{\text{sensor}}$ through the function C defined above (Fig. 2). The transformation of each line $L_i = (\mathbf{q_i}, \mathbf{v_i})$ in $\text{Ref}_{\text{sensor}}$ by $\mathbf{T}(\mathbf{p})$ into a line $l_i(\mathbf{p})$ in Ref_{3D} is given by

$$l_i(\mathbf{p}) = (\mathbf{T}(\mathbf{p})\ \mathbf{q}_i, \mathbf{T}(\mathbf{p})\mathbf{v}_i) = (\mathbf{r}_i(\mathbf{p}), \mathbf{u}_i(\mathbf{p})). \tag{5}$$

In the true pose $\mathbf{T^*}$, every matching line is tangent to the surface S of the object. We assume that for complex objects, in a large neighborhood of an initial estimate of $\mathbf{T}^{(0)}$, $\mathbf{T^*}$ is the only 3D pose of S leading to such a geometric configuration (this will later be confirmed in our experiments). Based on this important assumption, we design an algorithm which moves the surface S towards the matching lines (or equivalently, moves the rigid set of matching lines towards the surface) until S is in contact with all of these lines. Intuitively, the rigid set of matching lines evolves in a potential field that is zero on the surface and increases with distance from the surface S.

To formulate the matching problem as a least squares problem, we need a function which estimates the distance from any matching line to S. We first define the 3D *unsigned distance* between a point $\mathbf{r}$ and the surface S, $d_E(\mathbf{r})$, as the minimum Euclidean distance between $\mathbf{r}$ and all the points $\mathbf{s}_i$ of S,

$$d_E(\mathbf{q}) = \min_{\mathbf{s}_i} d(\mathbf{q}, \mathbf{s}_i) = d(\mathbf{q}, \mathbf{s}_{i_{\min}}), \tag{6}$$

where d is the Euclidean distance.

If we define the distance between a line $l_i(\mathbf{p})$ and the surface S as the minimum of $d_E(\mathbf{r})$ over all the points that belong to $l_i(\mathbf{p})$, then the distance associated with a line piercing the surface would be zero even though the line would not be tangent

to S. To avoid this problem, we define the *signed distance*, $\tilde{d}(\mathbf{r})$, between a point $\mathbf{r}$ and a closed surface S as $d_E(\mathbf{r})$ if $\mathbf{r}$ is exterior to S, and as the negative of this distance if $\mathbf{r}$ is interior to S. We thus define the signed distance between a line $l_i(\mathbf{p})$ represented in Ref_{3D} and the surface S, $\tilde{d}_l(l_i(\mathbf{p}))$, as the minimum of $\tilde{d}(\mathbf{r})$ over all the points that belong to $l_i(\mathbf{p})$.

$$\tilde{d}_l\big(l_i(p)\big) \equiv \min_{\lambda} \tilde{d}\big(\mathbf{r_i}(\mathbf{p}) + \lambda \mathbf{u_i}(\mathbf{p})\big). \tag{7}$$

Therefore, while the minimum unsigned distance along a line piercing the surface may be zero, the signed distance will be negative. Thus, whether the line $l_i(\mathbf{p})$ passes through the surface or not, since we look for the minimum value of the signed distance along $l_i(\mathbf{p})$, the absolute value of $\tilde{d}_l(l_i(\mathbf{p}))$ indicates whether the line $l_i(\mathbf{p})$ is *close* and *tangent* to S (Fig. 2).

When the final pose is reached, every line $l_i(\mathbf{p})$ is tangent to S, and every signed distance $\tilde{d}_l l_i(\mathbf{p}))$ is therefore zero. The energy minimized in our least squares model is

$$E(\mathbf{p}) = \sum_{i=1}^{M} \frac{1}{\sigma_i^2} \Big[\tilde{d}_l\big(l_i(\mathbf{p})\big)\Big]^2, \tag{8}$$

where σ_i^2 is the variance of the noise of the measurement $d_l(l_i(\mathbf{p}))$ (see Section VI). Given an initial estimate $\mathbf{p} = \mathbf{p}_0$ of the 3D/2D transformation parameters, we will use the Levenberg-Marquardt algorithm (Section VI) to perform the minimization. The next three sections detail the steps involved in this method.

IV. Fast Distance Computation Using Octree Splines

The method described in the previous section relies on the fast computation of the distances $\tilde{d}$ and d_l. The representation chosen for the input surface is a set of n points $\mathbf{s}_i$, $i = 1 \ldots n$. This choice is motivated by the fact that any surface representation can be converted into this point representation. Therefore, the computation of the distance $\tilde{d}$ is a $O(n)$ process. To speed up this process, we precompute a 3D *distance map*, which is a function that gives the signed minimum distance to S from any point $\mathbf{q}$ inside a bounding volume V that encloses S. In practice, a scale factor k is selected such that this bounding volume V is k times greater that the smallest cube that contains S.

We first studied and implemented uniform 3D distance maps [31]. In looking for an improved trade-off between memory space, accuracy, speed of computation, and speed of construction, we developed a new kind of distance map which we call the *octree spline*. The intuitive idea behind this geometrical representation is to have more detailed information (i.e., more accuracy) near the surface than far away from it. We start with the classical octree representation associated with the surface S [32] and then extend it to represent a continuous 3D function that approximates the signed Euclidean distance to the surface. This representation combines advantages of adaptive spline functions and hierarchical data structures. Starting from a set of n points $\mathbf{s}_i$ regularly spread on the surface S, the octree spline construction algorithm performs the following steps:

1) **Surface point octree construction:** First, the octree associated with the set of points $\mathbf{s}_i$ is built according to classic octree subdivision [32]. Starting from the initial cube V, each node that contains points (grey node) is recursively split into eight subcubes until it contains no points (white node) or it has the maximal selected resolution (black node). For our applications, the octree typically has six to nine levels, corresponding to a resolution of 64 to 512 cubes on a side.
2) **Subdivision (refinement):** The octree previously computed may have large empty nodes near the surface, because no rules about subdivision near the surface have been introduced. To overcome this problem, we perform a further subdivision of the octree to ensure that two nodes which are neighbors along a face, edge, or corner differ in size by at most a factor of k_S (in practice, we choose $k_S = 2$). In order to compute the 26 neighbors of any node in the octree, we use the technique described in [33]. Our refinement is then performed by traversing the octree in a bottom-up breadth-first fashion. The resulting structure is called a *restricted octree* [34].
3) **Corner distance computation:** For each corner $\mathbf{c}$ of each terminal node (white or black), the distance $d(\mathbf{c})$ is computed according to (6). The spatial organization of surface points created by the octree makes this process much faster since we can use best-first search to find this minimum distance quickly [8].
4) **Signed distance computation:** In order to compute $\tilde{d}$ from d_E, we use a two-step method. In the first pass, each $\tilde{d}(\mathbf{q})$ is set to $-\tilde{d}(\mathbf{q})$ using the result of step 3. Then, a positive sign is spread from an initial exterior point. This recursive spreading of the positive sign from points to their neighbors is terminated at points close to the surface.
5) **Crack elimination (continuity enforcement):** Because the signed distance is computed at any point $\mathbf{q}$ by an interpolation based on the eight corner values of the terminal node that contains $\mathbf{q}$ (see below), discontinuities or *cracks* can appear if we simply interpolate the true corner distance values. Several solutions can be used to suppress the cracks [32]. We chose the following process for its simplicity: if a corner $\mathbf{c}$ of a node N_1 of size s_1 lies on a face or an edge of another node N_2 of size $s_2 > s_1$, then the distance value of the corner $\mathbf{c}$ is simply replaced by the distance computed at $\mathbf{c}$ by interpolation inside N_2. This method is applied to the octree with a top-down breadth-first traversal. Note that this process introduces only a small loss of accuracy since the octree had been previously *restricted* at step 2.

After the previous steps have been performed, $\tilde{d}(\mathbf{q})$ can be computed for any point $\mathbf{q}$ by first finding the terminal node N that contains the point $\mathbf{q}$ (using classical binary search) and then using a trilinear interpolation of the eight corner values d_{ijk}. If $(u, v, w) \in [0,1] \times [0,1] \times [0,1]$ are the normalized co-

ordinates of $\mathbf{q}$ in the cube N,

$$\tilde{d}(\mathbf{q}) = \sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{k=0}^{1} b_i(u)b_j(v)b_k(w)\tilde{d}_{ijk} \quad (9)$$

with $b_l(t) = \delta_l\, t + (1 - \delta_l)(1 - t)$. We can compute the gradient $\nabla\tilde{d}(\mathbf{q})$ of the signed distance function by simply differentiating (9) with respect to u, v, and w. Because $\tilde{d}$ is only C^0, $\nabla\tilde{d}(\mathbf{q})$ is discontinuous on cube faces. However, these gradient discontinuities are relatively small and do not seem to affect the convergence of our iterative minimization algorithm.

One additional preprocessing step is required in order to speed up the line minimization algorithm presented in the next section:

6) **Lower bound estimation:** Compute for each node N of the octree a function B_N which is a lower bound on the function $\tilde{d}$ inside N,

$$\forall \mathbf{q} \in N,\ B_N(\mathbf{q}) \le \tilde{d}(\mathbf{q}). \quad (10)$$

A suitable choice for a bound function, with a good trade-off between simplicity and accuracy, is the linear approximation

$$B_N(\mathbf{q}) = au + bv + cw + d. \quad (11)$$

Because trilinear interpolation (9) is used to compute $\tilde{d}_l(\mathbf{q})$, the inequality (10) can be solved recursively (bottom-up) for (a, b, c, d) using a modified simplex algorithm.

V. Line to Surface Distance Minimization

This section describes how the signed distance from a line to the surface $\tilde{d}_l(l_i(p))$ defined in (7) can be rapidly computed using the octree spline. One possible approach to computing the minimum expressed in (7), which works for any representation of the distance map, would be to use classical one-dimensional functional minimization. Because the function $f(\lambda) = \tilde{d}(\mathbf{r} + \lambda\mathbf{u})$ may have several local minima for nonconvex shapes, global search techniques or exhaustive search may have to be used. This may be unreliable or prohibitively slow.

Our approach is to take advantage of the octree spline to develop an optimized search technique that requires approximately $O(\log(N))$ steps (where N is the resolution of the octree). We use best-first search with lower bounds computed for each node (interior and terminal) of the octree spline. Best-first search requires a priority queue, which we implement using a heap. In our algorithm, a candidate line $l_i(\mathbf{p})$ is recursively split by the octree cube boundaries into smaller line segments

$$\Sigma = [\boldsymbol{a}, \boldsymbol{b}]$$

until the line segment with minimum distance is found. During this process, the current minimum distance d_{best} is maintained, and an estimated lower bound on the distance

$$B_N(\Sigma) = \min(B_N(\mathbf{a}), B_N(\mathbf{b})) \quad (12)$$

is computed for each new segment before it is pushed onto the priority queue. As segments are popped from the queue in order of smallest lower bound, they are either split into smaller segments (if the associated node is non-terminal), or tested to see if the minimum attainable distance

$$\tilde{d}_l(\Sigma) = \min\left(\tilde{d}(\mathbf{a}), \tilde{d}(\mathbf{b})\right) \quad (13)$$

is lower than d_{best} (in which case $\mathbf{a}$ or $\mathbf{b}$ becomes the new minimum point). Note that while (13) is not the exact minimum distance over the line, the approximation is quite good, especially near the correct pose, where the line segments Σ are tangent to the surface S and the corresponding nodes N have the minimal size. The best-first search is terminated when the smallest lower bound on the priority queue is greater than d_{best}.

VI. Least Squares Minimization

This section describes the nonlinear least squares minimization of the energy or error function $E(\mathbf{p})$ defined in (8). Least squares techniques work well when we have many uncorrelated noisy measurements with a normal (Gaussian) distribution.[1] To begin with, we will make this assumption, even though noise actually comes from calibration errors, 2D and 3D segmentation errors, the approximation of the Euclidean distance by octree spline distance maps, and nonrigid displacement of the surface between Ref_{3D} and Ref_{sensor}.[2]

To perform the nonlinear least squares minimization, we use the Levenberg-Marquardt algorithm because of its good convergence properties [36]. Merging (5), (7), and (8), we get

$$E(\mathbf{p}) = \sum_{i=1}^{M}\frac{1}{\sigma_i^2}[e_i(\mathbf{p})]^2 = \sum_{i=1}^{M}\frac{1}{\sigma_i^2}\left[\min_{\lambda}\tilde{d}(\mathbf{T}(\mathbf{p})\mathbf{q}_i + \lambda\mathbf{T}(\mathbf{p})\mathbf{v}_i)\right]^2. \quad (14)$$

In order to compute the gradient and Hessian of $E(\mathbf{p})$, the Levenberg-Marquardt algorithm requires the first derivatives of each $e_i(\mathbf{p})$.[3] For any component p_j of $\mathbf{p}$, we obtain

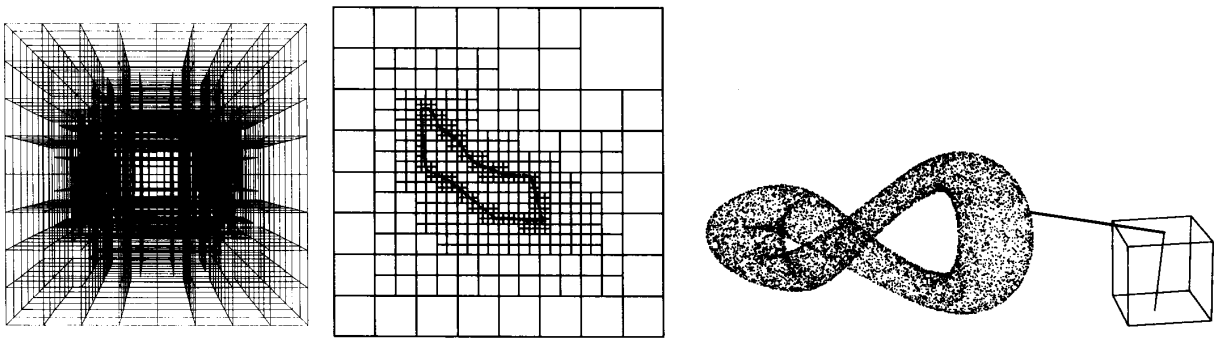
$$\frac{\partial e_i(\mathbf{p})}{\partial p_j} = \left[\nabla\tilde{d}(\mathbf{T}(\mathbf{p})\hat{\mathbf{q}}_i)\right]\cdot\left[\left(\frac{\partial\mathbf{T}(\mathbf{p})}{\partial p_j}\right)\hat{\mathbf{q}}_i\right], \quad (15)$$

where $\hat{\mathbf{q}}_i = \mathbf{q}_i + \lambda_{\min}\mathbf{v}_i$ and $\lambda_{\min}$ is the value of λ where the minimum is reached in (14). Computing the six-component gradient of $E(\mathbf{p})$ only requires computing the gradient of the octree spline distance $\tilde{d}$ (by differentiating (9)) and computing the three derivatives of $\mathbf{T}(\mathbf{p})$ with respect to the three Euler angles (the three derivatives of $\mathbf{T}(\mathbf{p})$ with respect to transla-

[1] Under these assumptions, the least squares criterion is equivalent to maximum likelihood estimation.

[2] In our application, 2D and 3D segmentation errors usually predominate since accurate sensor calibration is always feasible (Section III.B) and the resolution of the octree can be selected high enough to meet this requirement. A rough estimate of σ_i^2 in (8) can thus be obtained from $\sigma_{2D}^2+\sigma_{3D}^2$ where σ_{2D}^2 is the variance of the segmented contours on the projection images and σ_{3D}^2 is the variance of the 3D segmentation. These values can either be fixed to some fraction of the pixel/voxel size, or derived from the image gradient and the edge operator [35].

[3] Although it is not strictly necessary, we have found it beneficial in terms of accuracy, computation time, and stability of convergence to analytically compute the gradient of the error functions rather than performing approximations with finite differences.



(a) (b) (c)

Fig. 3. Octree frame and line to surface distance: (a) octree frame for surface S_2 (only five resolution levels are dislpayed), (b) 2D slice of the octree spline (surface S_1) showing the intersection between a plane interactively selected by the user and the octree spline, (c) line to surface distance computation: for an

tional components are simple constants).

The end of the iterative minimization process is reached when $E(\mathbf{p})$ is below a fixed threshold, when the norm of the gradient of $E(\mathbf{p})$ is below a fixed threshold, or when a maximum number of iterations is reached. At this point, we compute a robust estimate of the parameter $\mathbf{p}$ by throwing out the measurements where $e_i^2(\mathbf{p}) \gg \sigma_i^2$ and performing some more iterations [37]. This process removes the influence of the *outliers* which are likely to occur in the automatic 2D and 3D segmentation processes (for instance, a partially superimposed object on X-ray projections can lead to false contours). The threshold for outlier rejection must be fixed according to application-specific knowledge or by experimentation. In our case, we chose to set this threshold to 3σ where σ is a mean *a priori* standard deviation of the noise.

When using a gradient descent technique such as Levenberg-Marquardt, there is a possibility that the minimization might fail because of local minima in the six-dimensional parameter space. However, for the experiments we have conducted (Section VII.B), false local minima were few and always far away from the solution. Since in our application we have a good initial estimate of the parameters, these other minima are unlikely to be reached. Moreover, an initial least squares estimate of the translation parameters based on image moments allows us to avoid most of the false local minima far from the solution.

At the end of the iterative minimization procedure, we estimate the uncertainty in the parameters. The covariance matrix $\mathrm{Cov}(\mathbf{p})$ is computed by inverting the final Hessian matrix, and eigenvalue analysis of $\mathrm{Cov}(\mathbf{p})$ is performed [38]. The result is an ellipsoid of uncertainties in the six-dimensional parameter space, which could be used in further processing to quantify the uncertainty in other geometrical measures. For example, when we specify a 3D line in the data, we could also display its associated uncertainty envelope. The distribution of errors after minimization is also computed in order to check that it is Gaussian.

VII. Experimental Results

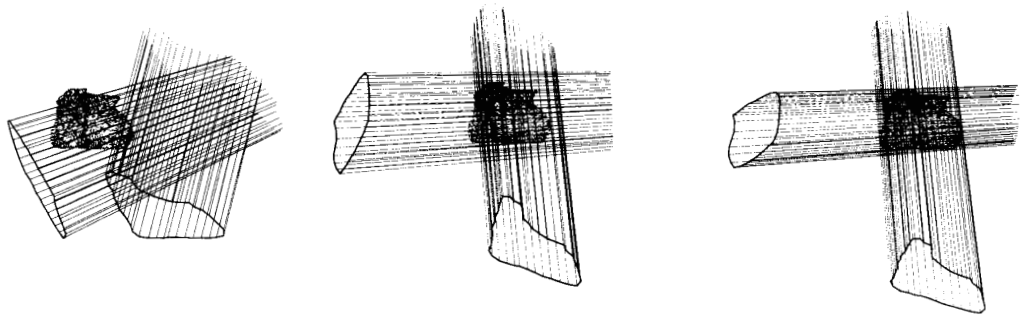
The octree spline construction and the matching algorithm have been implemented in C, using X Windows on a DECstation 5000/200. Computation times expressed below are given for this machine. For the experiments, we always use two 2D projections, oriented at right angles to each other (in real applications, the relative positions of the two sensors are carefully calibrated). The problem also has a unique solution if only one projection is used. However, in practice, a stable solution could be obtained with one projection only if highly accurate 2D and 3D segmented data were available. Typically, subpixel and subvoxel segmentation would be necessary for our applications.

A. Graphical Tools

As part of our implementation, we have developed a suite of real-time interactive 3D graphical tools which help us to visualize both the octree spline construction process and the iterative matching process (Figs. 3 and 4). For instance, 2D slices of the octree spline can be computed in real-time, and the quadtree decomposition of the slice or a pseudocolored rendering of the signed distance function $\tilde{d}$ can be displayed (Fig. 3(b)). The distance values $\tilde{d}$ (or $\tilde{d}_l$) can be interactively displayed for any point on a slice or along any interactively selected line (Fig. 3(c)).

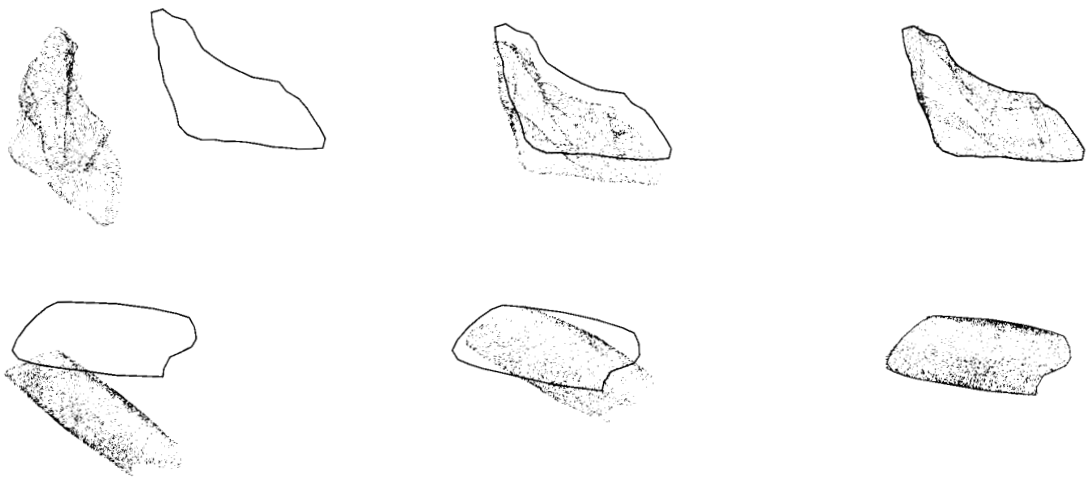
B. 3D/2D Matching with Simulated Sensor Data

This section describes our experiments with 3D/2D matching. We have performed tests on both real anatomical surfaces (surface S_1, Fig. 4) and on simulated surfaces (surface S_2, Fig. 6). The projection curves of these surfaces were obtained



(a) (b) (c)

Fig. 4. Convergence of the algorithm observed from a general viewpoint (surface S_1 is represented by a set of points). Two sets of projection lines evolve in the 3D potential field associated with the surface until each line is tangent to S_1: (a) initial configuration, (b) after two iterations, (c) after six iterations. For this case, the matching is performed in 1.8 s using 77 projection lines, in 0.9 s using 40 projection lines.



(a) (b) (c)

Fig. 5. Convergence of algorithm for surface S_1 observed from the two projection viewpoints. The external contours of the projected surface end up fitting the real contours: (a) initial configuration, (b) after two iterations, (c) after six iterations.

(a) (b) (c)

Fig. 6. Convergence of algorithm for surface S_2 (twisted torus) observed from a camera's viewpoint. For this case, the matching is performed in five seconds using 135 projection lines, in one second using 15 projection lines: (a) initial configuration, $E(\mathbf{p}^{(0)})/M = 63.87$, $|\Delta\mathbf{t}^{(0)}| = 44.10$ *mm*, $|\Delta\alpha^{(0)}| = 48.25°$, (b) after two iterations, $E(\mathbf{p}^{(2)})/M = 13.33$, $|\Delta\mathbf{t}^{(2)}| = 10.72$ *mm*, $|\Delta\alpha^{(2)}| = 14.75°$, (c) after 10 iterations. $E(\mathbf{p}^{(10)})/M = 0.91$, $|\Delta\mathbf{t}^{(10)}| = 0.21$ *mm*, $|\Delta\alpha^{(10)}| = 0.16°$.

by simulation in order to know "ground truth" for the parameters $\mathbf{p}^*$. The different steps of our experiments are the following:

1) Simulate a transformation $\mathbf{T}(\mathbf{p}^*)$ applied to S and compute the N silhouettes of the transformed surface $S'(\mathbf{p}^*)$ by projecting all the surface points.
2) Extract the N contours of the silhouettes of $S'(\mathbf{p}^*)$.
3) Randomly select M pixels on the N contours (a percentage of all the contour pixels is chosen, typically 10%, or 30–40 points). For each selected pixel compute the corresponding projection line according to the simulated projection parameters.
4) Compute the octree spline $\tilde{d}$ associated with the original surface S.
5) Starting from an initial estimate $\mathbf{p}^{(0)}$, use the Levenberg-Marquardt algorithm to iteratively minimize the function $E(\mathbf{p})$ defined by (14), using the gradient expressions given in (15). At each iteration k, compute and display the error $\Delta\mathbf{p} = \mathbf{p}^{(k)} - \mathbf{p}^*$ between the current parameters $\mathbf{p}^{(k)}$ and the simulated parameters $\mathbf{p}^*$. Compute the transformation error

$$\Delta\mathbf{T}^{(k)} = [\mathbf{T}(\mathbf{p}^{(k)})]\,[\mathbf{T}(\mathbf{p}^*)]^{-1} \tag{16}$$

and extract the translation error

$$\Delta\mathbf{t}^{(k)} = \mathbf{g} - \Delta\mathbf{T}^{(k)}\mathbf{g},$$

where $\mathbf{g}$ is the center of gravity of the surface points in Ref_{3D}. Extract $\Delta\alpha^{(k)}$, which is the angle of the rotation component $\mathbf{R}$ of $\Delta\mathbf{T}^{(k)}$. The values of $\|\Delta\mathbf{t}^{(k)}\|$ and $|\Delta\alpha^{(k)}|$ are displayed to monitor the convergence of the algorithm towards the optimal solution.
6) Perform robust estimation by removing the outliers and performing some more iterations.
7) Perform error analysis: compute the covariance matrix and the corresponding eigenvalues and eigenvectors; compute and display the error distribution.

Figs. 4 to 7 show the results of some of the experiments that we have conducted. Figs. 4 and 5 show the state of the iterative minimization algorithm for surface S_1 after 0, 2, and 6 iterations. Fig. 4 shows the relative positions of the projections lines and the surface seen from a general viewpoint. We can see how the projection lines become tangent to the surface as the algorithm recovers the correct pose. Fig. 5 shows the same state seen from the viewpoints of the two cameras. Here, we see how the surface represented by the cloud of dots completely fills the contour as the algorithm converges (this is equivalent to the tangency condition). Fig. 6 shows similar results for surface S_2. From this second example, we can see that the algorithm can deal with objects of very complicated shape.

A plot of the algorithm convergence (energy, translation error, and rotation error) is shown in Fig. 7. We see that the algorithm has essentially converged to its final value in just five steps. A plot of the region of convergence (number of steps to convergence vs. starting error) is shown in Fig. 8. Fig.8a was generated by varying each of the three Euler angles independently, and recording the number of steps required to converge. Fig. 8b was generated by varying the angles jointly (regular sampling of the 3D rotational parameter space), and collapsing the results onto a 1D plot based on the magnitude of the angular error. In these figures, only initial rotations were tested, since the translation is always initialized based on image moments (Section VI). As we can see from these figures, the algorithm always converges when started within 20° of the correct solution, and often converges from as far away as 60°. We can therefore conclude that for our applications, local minima are not a problem.

Table I shows some results for the computation times of $\tilde{d}_l$ and how they vary according to the choice of search strategy and octree resolution. As we can see from these results, the algorithm very quickly finds the optimal match between the surface and its projections. The best-first search algorithm with linear bounds on the distance function (11) performs better than either using a constant bound in each octree cell, or exhaustive (depth-first) search.

TABLE I
NUMBER OF NODES VISITED AND COMPUTATION TIMES (MSEC) FOR VARIOUS ALGORITHMS AND RESOLUTIONS •

octree resol.	const. bound nodes visited	const. bound time	linear bound nodes visited	linear bound time	depth first nodes visited	depth first time	error rot. / trans.
32	1494	195	1098	179	2827	243	1.02° / 0.25mm
64	2200	251	1353	212	4002	330	0.89° / 0.28mm
128	2800	381	1973	300	5371	455	0.32° / 0.43mm

Note. The computation times, obtained on a DECstation 5000/200, are per iteration, averaged over the last 10 iterations. The three algorithms tested are: (1) best-first search with constant lower bounds, (2) best-first search with linear lower bounds, (3) depth-first (exhaustive) search. The choice of algorithm does not affect the accuracy of the results (last column), which only depends on the resolution.

C. 3D/2D Matching with Real Sensor Data

The accuracy and robustness of our 3D/2D matching algorithm has also been verified on real data with two types of experiments.

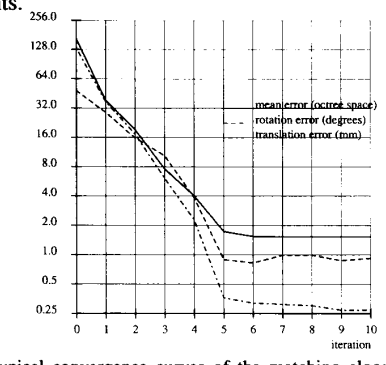
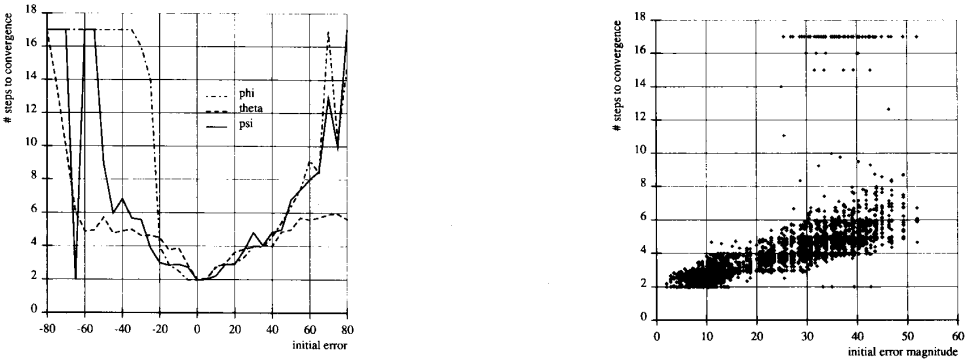


Fig. 7. Typical convergence curves of the matching algorithm (surface S_1) showing the mean error $E(\mathbf{p}^{(k)})/M$ (in voxels), translation error $\|\Delta\mathbf{t}^{(k)}\|$ (in mm), and rotation error $|\Delta\alpha^{(k)}|$ (in degrees) as a function of iteration number k.

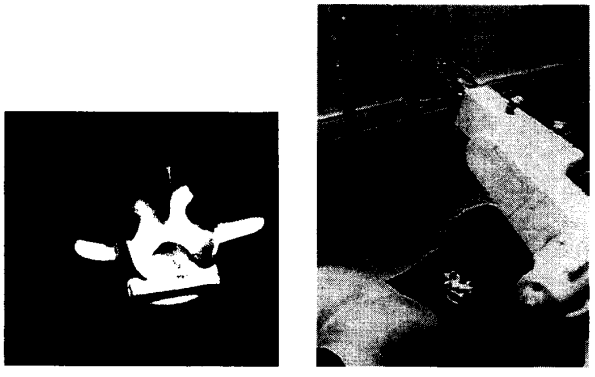


(a) (b)

Fig. 8. Typical convergence behavior for the matching algorithm (surface S_1). The plots show the number of iterations required to converge to within 5% of the final error as a function of the starting error in orientation: (a) for the three Euler angles and (b) for all orientations sampled on a rectangular grid. Data points above 14 indicate a failure to converge (false local minimum).

In the first type of experiment, we prove that the matching is correct using a complete robotics system. The test object used is an isolated vertebra which was pierced with two tubular 3-mm holes. A 3D CT scan of this vertebra was obtained and the positions of the hole axes $A1$ and $A2$ were computed. The 3D surface of the object was then segmented in the CT image. The result is a set of 200,000 points which is used to build a six-level octree spline.

For each experimental trial, the vertebra is placed in the fields of view of two calibrated cameras. Fig. 9a shows a typical image of the vertebra. A simple edge extraction algorithm gives a set of between 10 and 200 contour points on each image. The 3D/2D matching algorithm is then applied. Fig. 10 shows a typical convergence from one viewpoint. Note that the vertebra is partially occluded by the edge of the video camera so that only a few points are used in the minimization.



(a) (b)

Fig. 9. (a) Image of plastic vertebra used as data in the 3D/2D matching algorithm. (b) Photograph of robot experimental setup, with vertebra inside anatomical model, and robot holding the guiding laser.

At the end of the convergence, we obtain the final pose T. We use a six-axis robot carrying a laser beam and calibrated with the pair of cameras to align the laser beam with either hole axis $A1$ or $A2$ (Fig. 9(b)). We then visually determine if the laser beam has passed through the 3-mm hole. The observed result is that in all our experiments, the matching was visually perfect, which corresponds to a submillimeter accuracy. If we move the vertebra, 1 to 4 seconds are needed to perform a new matching and to move the robot (no human intervention is required).

A second type of experiment uses radio-opaque landmarks placed into a skull phantom (plastic model) for accuracy tests. The skull is segmented in the CT images, which gives us a 3D surface from which a seven-level octree-spline is built. The location of the centroid of four small catheters placed into the skull are also detected in the CT images. Then, two real calibrated X-rays of the skull are taken. The edges of the skull are manually segmented in the pair of X-rays. The centroids of the catheters are also detected in the X-rays. Therefore, we can reconstruct the 3D position of the 4 catheters in the sensor coordinate system by looking for the intersection of the pairs of corresponding projection lines. Now, applying our 3D/2D matching algorithm between the 3D surface and the two contours, we can measure the residual distances between the catheters given in Ref_{3D} and in Ref_{sensor}. Fig. 11 shows a typical convergence from one viewpoint. After convergence, the mean residual error on the four distances between catheters is 1.7 mm with a maximum of 2.3 mm. This accuracy is comparable to the errors induced by the interactive digitization of the catheters in the X-rays. This example shows that the method works accurately for shapes that are roughly spherical, without need for any high changes of curvature on the surface.

VIII. Discussion

This section discusses our matching algorithms and the geometrical representation we introduced for 3D distance maps.

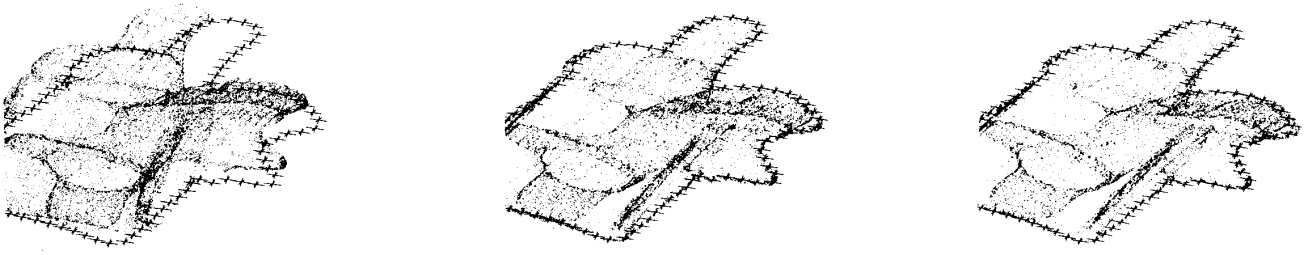
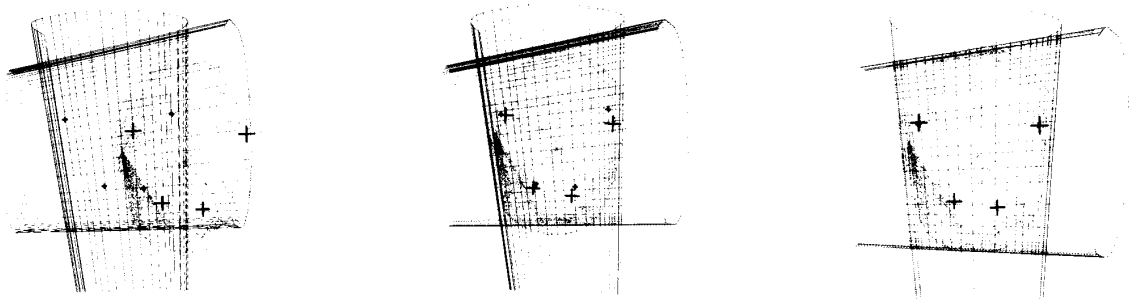


Fig. 10. Typical convergence for the vertebra observed from one viewpoint.



(a) (b) (c)

Fig. 11. Convergence of algorithm for a skull surface using two real X-rays (face and profile) observed from an arbitrary viewpoint. (a) initial configuration, (b) after one iteration, (c) after six iterations. Large crosses represent test catheters segmented on the 3D model, small crosses represent test catheters segmented on the X-ray images and reconstructed in 3D in the sensor coordinate system. In the final iteration (obtained in five seconds on a DECstation 5000/200), the mismatch between the four catheters is given by a mean distance of 1.7 mm.

A. *The Advantages of Matching by Distance Minimization*

This paper has presented a new algorithm for 3D surface to 2D contour matching based on nonlinear least squares. Our procedure performs a minimization of the 3D distance between the data (projection lines) and the surface. This approach is both simple and is particularly useful when no simple geometric description can be determined for the shape of the model,which is the case for anatomical structures. Our method is very general since it does not depend on any particular analytical representation of the surface (it just requires a collection of 3D data points). The simplicity of this problem formulation is enhanced by the fast computation of distances using the octree-spline representation.

The energy that we minimize corresponds to a sum of squared distances from each individual data point to the model and not vice versa. Thus, our algorithm can work with poor quality data provided that the accuracy of the model is high. Furthermore, the σ_i terms in equation (14) enable us to take into account uncertainties in the segmented data. Our experiments have demonstrated that only a small number of data points are necessary to obtain an accurate match. Finally, robust estimation allows us to deal with outliers. In applications where the data are X-ray images, this ability to perform matching with poor quality data is important, since edge extraction of a complete anatomical structure is often difficult.

A drawback of our global approach, compared with methods that rely on exhaustive combinatorial feature matching, is that local minima of the energy may exist. However, as mentioned previously, our method is intended to be used when a rough initial estimate of the object pose is known, which is the case in most medical applications. Moreover, our graphics software enables us to easily determine an initial pose for the object by interactively aligning the model with the data in 3D perspective views of the scene or in 2D projections of the scene. In applications where such approaches are not feasible, more work would be needed to develop a strategy for finding the global minimum [39].

In this paper, we have used our matching algorithms to estimate the pose of a known object. Our method could also be used for *recognition* problems, where the purpose is to match some data with a finite set of 3D objects $\{O_i\}$. After the robust estimation of the six attitude parameters $\mathbf{p}^*$ that link an object O_i with the sensed data, we can compute a *matching score* by looking at the residual error function $E(\mathbf{p}^*)$. The recognized object can be chosen as the one that minimizes this score [23]. For this approach to work, either a rough estimate of the ob-

ject's attitude would have to be known, or this would have to be estimated either by resorting to a global search of the parameter space, or through geometric hashing techniques [40].

B. The Advantages of Matching in 3D Space

This paper has developed a new algorithm for 3D surface to 2D contour matching, based on a minimization of the 3D distance between the projection lines and the surface. In earlier work, we had implemented a method that iteratively projected the surface S transformed by $\mathbf{T}(\mathbf{p})$ onto the image plane and minimized the 2D distance between the contours C_S of these simulated projections and the contours C_R segmented on the real input images. What, then, are the advantages of our new method with respect to this alternative approach (which is closer to traditional chamfer matching [27])?

First, computing the complete silhouette contours of the surface is more time consuming than looking for the minima of the octree spline along a small number of precomputed projection lines. This is especially true when projection models have to take local distortions into account.

Second, the distance measure between two arbitrary 2D contours that may only have partial overlap can be difficult to define and to minimize. Since fast computation is a requirement, a 2D distance map must be computed for each real contour projection [27]. However, 2D segmentation errors will be propagated through the distance map, which may lead to unpredictable results. Moreover, solving the minimization problem using the Levenberg-Marquardt algorithm requires the distances to be computed at points of the real edge, which implies the need to recompute distance maps at each iteration. In the technique of 2D distance minimization, gradient components cannot be computed analytically (while in our new method, we can use (15)). Approximation of the gradient by finite differences is thus required, which means computing six more projection contours at each iteration and weakening the convergence due to inaccurate gradients.

A potential weakness of our method is that we use only the external contour information in the projected images. It would be interesting to see how to add information about internal contours, variations of grey levels, differential properties to our matching process.

C. Other Applications of Octree Splines

The octree spline representation introduced in this paper has proven to be very effective at solving our matching problem. We are currently studying how to extend the octree spline and how to use it as a general modeling tool for a variety of applications.

First, we are studying how to construct octree splines from inputs more sophisticated than unordered point lists. Examples of such input surface representations include parametric splines, arbitrary triangulations, and level crossings in volumetric (voxel) data. Second, we are examining how to speed up the computation of the distance map $\tilde{d}$ at the corners of the octree,[4] using generalizations of chamfer distances to octrees as well as approximate techniques based on fast N-body solvers [41].

Finally, we are studying methods for extending the continuity of the octree spline from C^0 to C^1 and higher order continuities. The difficulty here is to merge overlapping spline basis functions with the octree representation. This new representation can be viewed as the combination of 3D adaptive meshes [42], hierarchical B-splines [43], and hierarchical data representations (pyramids) [44]. Compared with these other representations, octree splines have the potential to be less space consuming and to permit the rapid evaluation of certain geometric operations. Possible novel applications for the octree-spline include: obstacle avoidance in robotics using potential field methods; fast computation of intersections curves between two complex 3D surfaces; and surface interpolation of arbitrary point sets, using the zero-crossings of a C^k octree spline to implicitly represent the surface.

IX. Conclusions

In this paper, we have presented a new method for estimating the pose of a 3D smooth surface by matching it with a set of 2D projections. We have demonstrated the performance of this approach, coupled with accurate sensor calibration techniques, on a number of examples (simulations and real data). In the paper, we have assumed that 2D contours have been previously segmented. In practice, many methods exist for automated or semi-automated edge extraction from video images, but segmentation of X-ray contours still requires interactivity, since X-ray images represent complex scenes for which automated segmentation is difficult.

To our knowledge, the only other method that has been explicitly designed to solve the 3D/2D pose estimation problem for smooth surfaces is the work of Ponce and Kriegman [23]. However, their approach is different (it uses algebraic geometry) and it is not applicable to arbitrary-shaped surfaces such as the ones we consider. The main original contributions of our work are:

1) setting the 3D/2D matching problem in 3D instead of in 2D. We have defined a distance between 3D projection lines and a 3D surface that is zero when lines are tangent to the surface, such that the energy given by the sum of squares of line to surface distances is minimal when the surface is in the optimal pose.
2) the octree representation of the 3D distance map. This is the critical element of our method that makes the iterative matching feasible in practice, since line to surface distances can be computed very quickly.
3) using a simple and fast non-linear least-squares minimization technique—the Levenberg-Marquardt algorithm. In implementing this algorithm, we derived analytical expressions of the gradients of the energy, which makes the convergence fast and robust.

[4]This off-line preprocessing operation currently takes about 1.3 minutes on a DEC 5000/600 AXP for a 7-level octree with 40,000 points.

Using our novel technique, the four requirements presented in the introduction have all been met. First, the matching process works for any free-form smooth surface, since no *a priori* assumptions (symmetry hypotheses, algebraic forms, or special curves) are used. Second, we achieve a very high accuracy for the estimation of the six parameters in **p**, because the octree spline representation we use approximates the true 3D Euclidean distance with an error smaller than the segmentation errors in the input data. Third, we provide an estimate of the uncertainties of the six parameters, using least squares estimation with the Levenberg-Marquardt algorithm to compute these uncertainties. Fourth, we perform the matching process rapidly, using the octree spline to compute a signed distance from a line to the surface very quickly.

In current work, we are applying these techniques to robot-assisted surgery [2], [4], advanced 3D imaging and biomechanics [45] and 3D object localization applications [9]. We are also extending our approach to perform elastic matching between 3D surfaces [16].

ACKNOWLEDGMENTS

We are very grateful to Professor Philippe Cinquin for his contribution to the idea of setting the 3D/2D matching problem in 3D, and to Lionel Brunie, Ph.D., for his help on the physical interpretations and on uniform 3D distance maps. This research is financially supported by Digital Equipment Corporation.

REFERENCES

[1] S. Lavallée, "Registration for Computer Integrated Surgery: Methodology, state of the art," R.H. Taylor, S. Lavallée, G.C. Burdea, and R.W. Mosges, eds., *Computer Integrated Surgery,* Cambridge, MA: MIT Press, 1995.

[2] R.H. Taylor, S. Lavallée, G.C. Burdea, and R.W. Mosges, eds., *Computer Integrated Surgery*, Cambridge, MA: MIT Press, 1995.

[3] G. Champleboux, S. Lavallée, P. Sautot, and P. Cinquin, "Accurate calibration of cameras and range imaging sensors, the NPBS method," *IEEE Int'l Conf. on Robotics and Automation*, pp. 1552–1558, Nice France, May 1992.

[4] P. Sautot, P. Cinquin, S. Lavallée, and J. Troccaz, "Computer assisted spine surgery: A first step towards clinical application in orthopaedics," *14th IEEE Eng. in Medicine and Biology Conf.*, pp. 1071–1072, Paris, Nov. 1992.

[5] S. Leitner, I. Marque, S. Lavallée, and P. Cinquin, "Dynamic segmentation: Finding the edge with spline snakes," P.J. Laurent, ed., *Curves and Surfaces*, Chamonix, France: Academic Press, 1991.

[6] I. Marque, *Segmentation d'Images Medicales Tridimensionnelles Basée sur une Modelisation Continue du Volume*, PhD thesis, Grenoble University, France, Dec. 1990.

[7] O. Monga, R. Deriche, G. Malandain, and J. P. Cocquerez, "Recursive filtering and edge closing: Two primary tools for 3D edge detection," *First European Conf. on Computer Vision*, pp. 56–65, Antibes, France: Springer-Verlag, April 1990.

[8] S. Lavallée, R. Szeliski, and L. Brunie, "Matching 3D smooth surfaces with their 2D projections using 3D distance maps," In *SPIE Vol. 1570, Geometric Methods in Computer Vision*, pp. 322–336, San Diego, CA, July 1991.

[9] G. Champleboux, S. Lavallée, R. Szeliski, and L. Brunie, "From accurate range imaging sensor calibration to accurate model-based 3D object localization," *IEEE CS Conf. on Comput. Vision and Pattern Recognition (CVPR '92)*, Champaign, Ill., June 1992.

[10] B.A. Kall, P.J. Kelly, and S.J. Goerss, "Comprehensive computer-assisted data collection treatment planning and interactive surgery," *SPIE, Medical Imaging, vol. 767*, pp. 27–35, 1987.

[11] C. Schiers, U. Tiede, and K.H. Hohne, "Interactive 3D registration of image volumes from different sources," H.U. Lemke, ed., *Computer Assisted Radiology,* Berlin: Springer-Verlag, pp. 667–669, June 1989.

[12] C.A. Pelizzari, G.T.Y. Chen, D.R. Spelbring, R.R. Weichselbaum, and C-T. Chen, "Accurate 3D registration of CT, PET, and-or MR images of the brain," *J. Computer Assisted Tomography*, 13(1):20–26, 1989.

[13] A. Guéziec and N. Ayache, "Smoothing and matching of 3D space curves," G. Sandini, ed., *Second European Conference on Computer Vision*, Santa Margherita, Italy, New York: Springer Verlag, LNCS Series Vol. 588, pp. 620–629, May 1992.

[14] D. Terzopoulos, A. Witkin, and M. Kass, "Constraints on deformable models: Recovering 3D shape and nonrigid motion," *Artificial Intelligence*, 36:91–123, 1988.

[15] R. Bajcsy and S. Kovacic, "Multiresolution elastic matching," *Computer Vision, Graphics, and Image Processing*, 46:1–21, 1989.

[16] R. Szeliski and S. Lavallée, "Matching 3D anatomical surfaces with non-rigid deformations using octree-splines," *IEEE Workshop on Biomedical Image Analysis*, Seattle, WA: pp. 144-153, June 1994.

[17] Y. Demazeau, *Niveaux de représentation pour la vision par ordinateur; indices d'image et indices de scène,* PhD thesis, INPG, Grenoble University, 1986.

[18] D. G. Lowe, *Perceptual Organization and Visual Recognition*, Boston, MA: Kluwer Academic Publishers, 1985.

[19] M. Dhome, M. Richetin, J. T. Lapreste, and G. Rives, "Determination of the attitude of 3D objects from a single perspective view," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, no. 12, pp. 1265–1278, 1989.

[20] C. Granger, *Reconnaissance d'objets par mise en correspondance en vision par ordinateur,* Nice University, France, 1985.

[21] D. G. Lowe, "Fitting parameterized three-dimensional models to images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 441–450, May 1991.

[22] P.J. Besl and N.D. McKay, "A method for registration of 3D shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[23] D. J. Kriegman and J. Ponce, "On recognizing and positioning curved 3D objects from image contours," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 12, pp. 1127–1137, December 1990.

[24] R. Glachet, M. Dhome, and J. T. Lapreste, "Finding the pose of an object of revolution," G. Sandini, ed., *Second European Conference on Computer Vision*, Santa Margherita, Italy, New York: Springer Verlag, LNCS Series Vol. 588, pp. 681–686, May 1992.

[25] R. Brooks, "Symbolic reasoning among 3D models and 2D images," *Artificial Intelligence*, 17:285–348, 1981.

[26] S. Sullivan, L. Sanford, and J. Ponce, "Using Geometric Distance Fits for 3D Object Modeling and Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 16, no. 12, pp. 1183-1196, 1994.

[27] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf, "Parametric correspondence and chamfer matching: Two new techniques for image matching," *Fifth International Joint Conference on Artificial Intelligence*, Cambridge, MA: pp. 659–663, August 1977.

[28] N. Ayache, *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception,* Cambridge, MA: MIT Press, 1991.

[29] R. Y. Tsai, "Synopsis of recent progress on camera calibration for 3D machine vision," *The Robotics Review*, Cambridge, MA: MIT Press, pp. 147–160, 1989.

[30] H. A. Martins, J. R. Birk, and R. B. Kelley, "Camera models based on data from two calibration planes," *Computer Graphics and Image Processing*, 17:173–179, 1981.

[31] G. Borgefors, "Distance transformations in arbitrary dimensions," *Computer Vision, Graphics, and Image Processing*, 27:321–345, 1984.

[32] H. Samet, *The Design and Analysis of Spatial Data Structures,* Read-

ing, MA: Addison-Wesley, 1989.
[33] G. Garcia, *Contribution a la modelisation d'objets et a la détection de collisions en robotique a l'aide d'arbres octaux*, PhD thesis, Nantes University, September 1989.
[34] B.V. Herzen and A.H. Barr, "Accurate triangulations of deformed, intersecting surfaces," *Computer Graphics*, 21(4):103–110, 1987.
[35] O. Monga, N. Ayache, and P. T. Sander, "From voxel to curvature," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Maui, HI, pp. 644–649, June 1991.
[36] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge, England: Cambridge Univ. Press, 1986.
[37] P. J. Huber, *Robust Statistics*, New York: John Wiley & Sons, 1981.
[38] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge, England: Cambridge Univ. Press, second ed., 1992.
[39] W.E.L. Grimson et al., "An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seattle, WA: pp. 430–436, June 1994.
[40] H. Wolfson, "On curve matching," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 5, pp. 483–489, May 1990.
[41] J. Barnes and P. Hut, "A hierarchical $O(N \log N)$ force-calculation algorithm," *Nature*, 324:446–449, December 4, 1986.
[42] D. Terzopoulos and M. Vasilescu, "Sampling and reconstruction with adaptive meshes," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Maui, HI, pp. 70–75, June 1991.
[43] D.R. Forsey and R.H. Bartels, "Hierarchical B-spline refinement," *Computer Graphics (SIGGRAPH '88)*, 22(4):205–212, August 1988.
[44] R. Szeliski, "Fast surface interpolation using hierarchical basis functions," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 6, pp. 513–528, June 1990.
[45] L. Brunie, "Accurate 3D modelization of the scoliosis by matching 3D images (MR, CT) with 2D X-rays," J. Dansereau, ed., *Int. Symp. on 3D Scoliotic Deformities*, Montreal, Canada: Gustav Fischer Verlag, pp. 11–17, June 1992.

Stéphane Lavallée received the engineering degree from Ecole Nationale Supérieure des Telécommunications de Bretagne in 1986 and the PhD degree in Biomedical Engineering from Grenoble University in 1989. In 1991, he worked at the Cambridge Research Laboratory of Digital Equipment Corporation in Cambridge, Mass. Since 1991, he has held a research position at the CNRS, in the TIMC (Techniques de l'Imagerie, la Modélisation et la Cognition) laboratory in Grenoble, France. His current research interests include registration of multimodality images and models, registration of images with surgical physical spaces, sensor calibration, and robot calibration.

Richard Szeliski received the BEng degree in Honours Electrical Engineering from McGill University, Montreal, in 1979, the MASc degree in Electrical Engineering from the University of British Columbia, Vancouver, in 1981, and the PhD degree in Computer Science from Carnegie Mellon University, Pittsburgh, in 1988. Since 1989 he has been a Member of Research Staff at the Cambridge Research Lab of Digital Equipment Corporation, Cambridge, where he is pursuing research in 3D computer vision, geometric modeling, medical image registration, and parallel programming and algorithms.